Claim Amendments:

1.   (Original) A data-sharing method wherein a non-
cooperative DBMS of a primary computer system participates in
unaware applications and has a cache, respective lock structures,
database log files and database data files responsive to data
requests generated by the unaware applications, said method
comprising the steps of:

(a) nonintrusively monitoring data written to said
database log files and said database data files and communicating
information as to data written to said files to a secondary DBMS
running on a potentially different computer and having a secondary
cache and secondary lock requests and responsive to data requests
by other unaware applications; and

(b) processing data in said secondary DBMS between said
other unaware applications and with said secondary cache and said
secondary lock requests while reading data from said non-
cooperative DBMS data files without interrupting update or
retrieval activities of said non-cooperative DBMS and while
isolating said non-cooperative DBMS from said other applications,

19   thereby enabling said other unaware applications to access the

20   data maintained by said non-cooperative DBMS.

1   2.   (Currently amended)   A data-sharing method wherein a

2   non-cooperative DBMS of a primary computer system participates in

3   unaware applications and has a cache, respective lock structures,

4   database log files and database data files responsive to data

5   requests generated by the unaware applications, said method

6   comprising the steps of:

7   (a) nonintrusively monitoring data written to said

8   database log files and said database data files and communicating

9   information as to data written to said files to a secondary DBMS

10   running on a potentially different computer and having a secondary

11   cache and secondary lock requests and responsive to data requests

12   by other unaware applications;

13   (b) processing data in said secondary DBMS between said

14   other unaware applications and with said secondary cache and said

15   secondary lock requests while reading data from said non-

16   cooperative DBMS data files without interrupting update or

17   retrieval activities of said non-cooperative DBMS and while

18   isolating said non-cooperative DBMS from said other applications,

19   thereby enabling said other unaware applications to access the

20   data maintained by said non-cooperative DBMS; and ~~The method~~

21   ~~defined in claim 1, further comprising the step of~~

22   (c) intercepting data written by said primary computer

23   system to said non-cooperative DBMS and parsing the intercepted

24   data nonintrusively with respect to said primary computer system

25   with a listener and utilizing the parsed intercepted data to

26   establish said secondary cache and secondary lock requests

27   shielding said other unaware applications from inconsistent data of

28   said primary computer system.

1   3.   (Currently amended)   A data-sharing method wherein a

2   non-cooperative DBMS of a primary computer system participates in

3   unaware applications and has a cache, respective lock structures,

4   database log files and database data files responsive to data

5   requests generated by the unaware applications, said method

6   comprising the steps of:

7   (a) nonintrusively monitoring data written to said

8   database log files and said database data files and communicating

9   information as to data written to said files to a secondary DBMS

10 running on a potentially different computer and having a secondary

11 cache and secondary lock requests and responsive to data requests

12 by other unaware applications; and

13   (b) processing data in said secondary DBMS between said

14 other unaware applications and with said secondary cache and said

15 secondary lock requests while reading data from said non-

16 cooperative DBMS data files without interrupting update or

17 retrieval activities of said non-cooperative DBMS and while

18 isolating said non-cooperative DBMS from said other applications,

19 thereby enabling said other unaware applications to access the data

20 maintained by said non-cooperative DBMS; ~~The method defined in~~

21 ~~claim 1wherein~~ said secondary DBMS ~~operates~~ operating with items of

22 interest having a structure consisting of a part defining an item

23 type distinguishing between parts of a data base, a part defining

24 an identity of the item in the database, a "dirty" part describing

25 parts of an item not previously transferred to storage, a part

26 describing a previous transaction involving the item to permit

27 updating of that transaction, a part describing a locking

28 transaction, a part facilitating application of an optimization

29   algorithm, a list of pending reads identifying processes which have

30   shown interest in the item, a part representing a before image

31   constituting a pointer to data represented by the item before the

32   transaction, a part representing an after image of data subsequent

33   to the transaction and a part representing a transaction initiated

34   by a respective one of said other unaware applications.

1   4.   (Original) The method defined in claim 3 wherein each

2   transaction with an item of interest is effected in said listener

3   by the steps of:

4   (i) check whether the item of interest is a first item of

5   the transaction;

6   (ii) if the item of interest is the first item of the

7   transaction, process the item of interest by exclusively locking

8   the transaction identification of the item of interest and creating

9   a transaction entry therefore;

10   (iii) if the item of interest is not the first item of

11   the transaction or after the creation of the transaction entry,

12   check whether the item of interest is already in cache;

13   (iv) if the item of interest is not already in cache,

14   create a cache entry therefore;

15   (v) if the item of interest is already in cache, check

16   whether the item of interest belongs to a previous transaction;

17   (vi) if the item of interest is already in cache but does

18   not belong to a previous transaction and following step (iv),

19   concatenate the cache entry with a transaction context and create

20   the before image for the item of interest; and

21   (vii) following step (vi) and, where the item of interest

22   following step (v) belongs to a previous transaction, updating the

23   cache entry to contain a new after image and "dirty" part.

1   5.   (Original) The method defined in claim 3, further

2   comprising checking an item of interest read by said listener in

3   the past by the steps of:

4   (I) checking whether the item of interest is locked by a

5   transaction;

6   (II) if the item of interest is not locked by a

7   transaction, verifying if the previous transaction for the item of

8   interest is the same as the previous transaction for a prior

9   reading by comparing the previous transaction part of the item of

10 interest with a corresponding entry of the previous transaction at

11 the prior reading; and

12   (III) validating the item of interest when the previous

13 transaction part is the same as the corresponding entry of the

14 previous transaction at the prior reading.

1   6.   (Original) The method defined in claim 3, further

2   comprising initiating at time intervals a post listener sequence in

3   said listener which comprises the steps of:

4   (A) scanning an item of interest cache entry;

5   (B) selecting entries of items of interest having NULL

6   locking transaction parts;

7   (C) for each item of interest having a NULL locking

8   transaction part, checking whether the item of interest entry has a

9   "dirty" part;

10   (D) for each item of interest found to have a "dirty"

11 part in step (C), reading corresponding data from storage and

12 updating the "dirty" part data; and

13   (E) following step (A) in the case of a cache entry of an

14 item of interest having a not NULL locking transaction part,

15   following step (C) for each cache entry having no "dirty" part and

16   following step (D) for each cache entry having an updated "dirty"

17   part, returning to step (A) to scan a next item of interest cache

18   entry until all item of interest cache entries are scanned.

1   7.   (Original)   The method defined in claim 3 wherein

2   update operations initiated in said secondary computer system are

3   delegated to the non-cooperative DBMS by the steps of:

4   transmitting from said secondary DBMS to said non-cooperative DBMS

5   of said primary computer system an update instruction based upon a

6   read transaction of one of said other unaware applications;

7   checking whether the update of the non-cooperative DBMS of the

8   primary computer system has been completed;

9   thereafter locking the transaction initiated by said one of said

10   other unaware applications; and

11   creating cache entries including after images for all items of

12   interest affected by the update.

1   8.   (Original)   The method defined in claim 2, further

2   comprising the step of updating, with said secondary DBMS in

3   response to one of said other unaware applications, the non-

4   cooperative DBMS at least in part by delegating update operations

5   and subsequent retrieval operations directly or indirectly to said

6   non-cooperative DBMS.

1   9.   (Currently amended)   The method defined in claim 1 2

2   wherein the cache associated with said secondary DBMS is provided

3   with a storage capacity sufficient to hold the entire contents of

4   the data of at least one of said DBMS.

1   10. (Currently amended)   A data-sharing method wherein a

2   non-cooperative DBMS of a primary computer system participates in

3   unaware applications and has a cache, respective lock structures,

4   database log files and database data files responsive to data

5   requests generated by the unaware applications, said method

6   comprising the steps of:

7   (a) nonintrusively monitoring data written to said

8   database log files and said database data files and communicating

9   information as to data written to said files to a secondary DBMS

10 running on a potentially different computer and having a secondary

11   cache and secondary lock requests and responsive to data requests

12   by other unaware applications; and

13   (b) processing data in said secondary DBMS between said

14   other unaware applications and with said secondary cache and said

15   secondary lock requests while reading data from said non-

16   cooperative DBMS data files without interrupting update or

17   retrieval activities of said non-cooperative DBMS and while

18   isolating said non-cooperative DBMS from said other applications,

19   thereby enabling said other unaware applications to access the

20   data maintained by said non-cooperative DBMS; ~~The method defined~~

21   ~~in claim 1 wherein~~ said primary computer system ~~has~~ having a

22   controller, said method further comprising the step of speeding a

23   response time of said non-cooperative DBMS and reducing an I/O load

24   on said controller by intercepting and eliminating the physical

25   write operations directed to a database of the non-cooperative

26   DBMS, and utilizing said controller to intercept log writes to

27   automatically trigger writes to a disk of the non-cooperative DBMS,

28   and for directly responding to read requests relying on data cached

29   in memory of said non-cooperative DBMS.

1   11. (Currently amended) The method defined in claim ~~1~~ <u>2</u>

2   wherein at least one of said DBMSs is provided with a time function

3   keeping a transaction time for an entire duration of each

4   transaction and creating an appearance of a transaction occurring

5   at a single point in time, thereby supporting true repeatable read

6   and serializable transactions.

1   12. (Currently amended) <u>A data-sharing method wherein a</u>

2   <u>non-cooperative DBMS of a primary computer system participates in</u>

3   <u>unaware applications and has a cache, respective lock structures,</u>

4   <u>database log files and database data files responsive to data</u>

5   <u>requests generated by the unaware applications, said method</u>

6   <u>comprising the steps of:</u>

7   <u>(a) nonintrusively monitoring data written to said</u>

8   <u>database log files and said database data files and communicating</u>

9   <u>information as to data written to said files to a secondary DBMS</u>

10 <u>running on a potentially different computer and having a secondary</u>

11 <u>cache and secondary lock requests and responsive to data requests</u>

12 <u>by other unaware applications;</u>

13   <u>(b) processing data in said secondary DBMS between said</u>

14   other unaware applications and with said secondary cache and said

15   secondary lock requests while reading data from said non-

16   cooperative DBMS data files without interrupting update or

17   retrieval activities of said non-cooperative DBMS and while

18   isolating said non-cooperative DBMS from said other applications,

19   thereby enabling said other unaware applications to access the data

20   maintained by said non-cooperative DBMS; and at least one of said

21   DBMSs being provided with a time function keeping a transaction

22   time for an entire duration of each transaction and creating an

23   appearance of a transaction occurring at a single point in time,

24   thereby supporting true repeatable read and serializable

25   transactions, and ~~The method defined in claim 11~~ wherein, for each

26   transaction requiring a "repeatable read" or "serializable"

27   isolation level, a transaction snapshot entry is created which

28   includes a transaction identifier identifying the transaction, a

29   log position identifier containing a last position in the log that

30   has been read at the instant the transaction is initiated, a time

31   stamp serving for date-related functions required by the logic for

32   the transaction and a process identifier pointing to a process

33   which issued the transaction.

1   13.   (Original)   A computer system comprised of a

2   secondary DBMS, a secondary cache and secondary lock structures and

3   connectable for data sharing with a non-cooperative DBMS of a

4   primary computer which participates in unaware applications and has

5   a cache, respective lock structures, database log files and

6   database data files responsive to data requests generated by the

7   unaware applications, said computer system having a listener

8   connected to said non-cooperative DBMS for:

9   (a) nonintrusively monitoring data written to said

10 database log files and communicating information as to data written

11 to said files to said secondary DBMS of said computer system, said

12 computer system being responsive to data requests by other unaware

13 applications; and

14   (b) processing data in said secondary DBMS between said

15 other unaware applications and with said secondary cache and said

16 secondary lock requests while reading data from said non-

17 cooperative DBMS with said secondary DBMS without interrupting

18 update or retrieval activities of said non-cooperative DBMS and

19 while isolating said non-cooperative DBMS from said other

- 14 -

20 applications, thereby enabling said other unaware applications to

21 access the data maintained by the non-cooperative DBMS.

1   14. (Original) The computer system defined in claim 13

2   wherein said listener, said secondary DBMS and said secondary cache

3   are provided in a single hardware unit separately from a computer

4   on which said other applications are run.

1   15. (Original) The computer system defined in claim 13

2   wherein said listener, said secondary DBMS and said secondary cache

3   are provided in a computer in which said other applications are

4   run.

1   16. (Currently amended) A computer system comprised of

2   a secondary DBMS, a secondary cache and secondary lock structures

3   and connectable for data sharing with a non-cooperative DBMS of a

4   primary computer which participates in unaware applications and has

5   a cache, respective lock structures, database log files and

6   database data files responsive to data requests generated by the

7   unaware applications, said computer system having a listener

8   connected to said non-cooperative DBMS for:

9   (a) nonintrusively monitoring data written to said

10   database log files and communicating information as to data written

11   to said files to said secondary DBMS of said computer system, said

12   computer system being responsive to data requests by other unaware

13   applications; and

14   (b) processing data in said secondary DBMS between said

15   other unaware applications and with said secondary cache and said

16   secondary lock requests while reading data from said non-

17   cooperative DBMS with said secondary DBMS without interrupting

18   update or retrieval activities of said non-cooperative DBMS and

19   while isolating said non-cooperative DBMS from said other

20   applications, thereby enabling said other unaware applications to

21   access the data maintained by the non-cooperative DBMS ~~The~~

22   ~~computer system defined in claim 13 wherein~~ said listener ~~is~~ being

23   connected to intercepting data written by said primary computer to

24   said non-cooperative DBMS and is programmed to parse the

25   intercepted data nonintrusively with respect to said primary

26   computer and utilize the parsed intercepted data to establish said

27   secondary cache and secondary lock requests shielding unaware

28   applications executed in said computer system from inconsistent

29   data of said primary computer.

1   17.   (Currently amended)   The computer system defined in

2   claim ~~13~~ 16 wherein said secondary cache is provided in a memory

3   with a storage capacity corresponding to the entire database of the

4   non-cooperative DBMS.

1   18.   (Currently amended)   A computer system comprised of

2   a secondary DBMS, a secondary cache and secondary lock structures

3   and connectable for data sharing with a non-cooperative DBMS of a

4   primary computer which participates in unaware applications and has

5   a cache, respective lock structures, database log files and

6   database data files responsive to data requests generated by the

7   unaware applications, said computer system having a listener

8   connected to said non-cooperative DBMS for:

9   (a) nonintrusively monitoring data written to said

10   database log files and communicating information as to data written

11   to said files to said secondary DBMS of said computer system, said

12   computer system being responsive to data requests by other unaware

13   applications; and

14   (b) processing data in said secondary DBMS between said

15   other unaware applications and with said secondary cache and said

16   secondary lock requests while reading data from said non-

17   cooperative DBMS with said secondary DBMS without interrupting

18   update or retrieval activities of said non-cooperative DBMS and

19   while isolating said non-cooperative DBMS from said other

20   applications, thereby enabling said other unaware applications to

21   access the data maintained by the non-cooperative DBMS, said

22   listener, said secondary DBMS and said secondary cache being

23   provided in a single hardware unit separately from a computer on

24   which said other applications are run ~~The computer system defined~~

25   ~~in claim 14 wherein~~ said listener and said secondary DBMS ~~are~~ being

26   combined with a storage controller responsive to SQL select and

27   other data manipulation read commands retrieving data of said non-

28   cooperative DBMS.

1   19.   (Original)   The computer system defined in claim 18

2   wherein at least one of said computers has a multiplicity of

3   dedicated CPUs for executing the SQL select and other data

4   manipulation read commands in parallel on different parts of the

5   secondary cache.

       Claim 20, cancelled.